Policy Driven Dynamic LUN Space Optimization based on the Utilization

Nayaka B Govindaraja [1], Majeed Zameer [1], Taranisen Mohanta[1]
[1] Hewlett-Packard , Bangalore, India
Email: {govindaraja.nayaka-b, zameer.majeed, taranisen.mohanta}@hp.com

***Abstract*—In a typical SAN solution consisting of intelligent storage system there will be many LUNs with pre-allocated space (thick LUNs) based on the business requirements. In case of thin provisioned LUNs physical space is not allocated upfront; space will be allocated on demand depending on incoming write workload. In thin provisioned LUN implementation some space will be anchored or pre-allocated and in other implementations no space will be anchored (everything is allocated on demand). Most often administrator oversubscribes the space for thick LUN's. In case of thin LUN's with anchored space some space will be reserved to accommodate data or incoming writes. There will be many oversubscribed thick and anchored thin LUNs in a storage systems resulting in non optimal usage of storage space.**

For some of the business needs oversubscribing space for thick and anchored thin LUN's without considering the utilization doesn't augur well. There should be a way to optimize storage space in an intelligent storage system based on LUN utilization over a period of time. By determining utilization of a LUN from time to time it's possible to have dynamic provisioning mechanism for thick and anchored thin LUNs based on the usage over a time.

The proposed policy driven thick and anchored thin LUN optimization will help storage admin to optimize space in a storage system.

***Index Terms*-SAN, LUN, Thin provisioning,anchored,policy**

I. Introduction

The need of reliable storage is essential in the storage array is very essential [2]. To scale up the business need thin provisioning LUNs are very demanding in current data centers to cater the I/Os [1]. The disks in the disk arrays tries to cater more performance [4][5][6]. The storage administrator has to maintain the storage array to perform optimally [3]. Currently there are no methods to determine set of underutilized LUNs over a period of time "T" to optimally utilize storage space in a storage system. The proposed solution determines the list of underutilized LUNs in a storage system and releases space based on LUN utilization over a period of time "T" by taking user specified policy parameters. It determines best utilized LUNs over a period of time and adds space to best utilized thin LUNs in a storage system.

There will be different user provided policy parameters will be used. If user doesn't provide the policy parameters then default values will be considered.

II. Problems Solved

In a typical SAN solution administrator or User often cre ates thick and anchored thin LUN's with much more capacity or space than actually needed. Because of this there will be plethora of thick and anchored thin LUN's in a storage system with overcommitted space and hence space in a storage system itself is not optimal. Since there are many LUNs with overcommitted space it's not possible to accommodate new LUN's in a storage system and there is no way to determine the list of underutilized LUN's. Currently there are no methods available to predicatively determine underutilized LUN's over a period of time "T", release space from them and convert them from thick to thin LUN based on the utilization.

Finding the list of underutilized LUNs and releasing space from them according to the utilization of each LUN will help admin to optimize space in a storage system. By having a list of underutilized LUNs it's possible to assign a ranking to each LUN, form a ranking table and release free space from each LUN according to the rank. Each thick LUN becomes a thin LUN with new free capacity. Each thin LUN with non-zero free capacity assumes a new value for a free capacity once space is released from free space.

III. Prior Solution

One way to solve aforementioned problem is to convert overcommitted thick LUN's to thin LUN's with no anchored or pre-allocated space. When thick LUN is converted to a thin LUN all the available free space in a LUN will be freed and space for incoming writes will be allocated on demand. However in a system with lot of incoming write workload on demand allocation doesn't perform better compared to thick LUN's with pre-allocated space or thin LUNs with anchored or pre-allocated space. Also, converting each thick LUN to thin LUN without considering the utilization or incoming write workload might result in performance degradation because of on demand allocation.

IV. Proposed Solution

As shown in the figure 1 there are several storage arrays connected to a fabric. Host is connected to a fabric and it sees all the LUN's presented by a particular storage array. There is a management station connected to a fabric to manage a storage array which provides a GUI/CLI. A high availability repository is also connected to a fabric and management station uses this repository to write hourly usage of each LUN in a storage array. Using management station it's possible to log either hourly usage of all the LUNs in a particular storage array or hourly usage of all the LUN's in every

DOI: 03.LSCS.2013.1.513

storage array connected to a fabric. An application having business logic to determine the utilization of a LUN runs in management station and has the ability to dynamically provision LUN's based on utilization over a time.

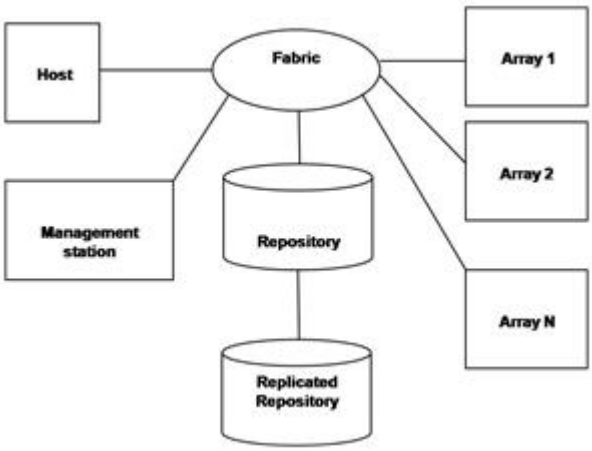

Figure 1. Proposed architecture

In the proposed solution communication between various entities happen as below as per flow in Figure 2:

1. Management station (or an application) queries storage array to get hourly available space for each LUN. Storage array keeps track of hourly available space for each LUN in a non-volatile counter. Management stations continuously polls the storage array on hourly basis to get the relevant data.
2. Management station updates in hourly basis available space of LUNs in a storage array to repository. Ensures to synchronize the storage array data and repository.
3. Based on the user policy parameters an application which in management station having business logic to determine the LUN utilization runs on a weekly or monthly basis. From the repository data determines worst and best utilized LUN's over a period of time "T".
4. Management station sends a request to storage array to release some space from the worst utilized LUN's based on user specified or configured parameter. Space will be released from available space (or free space) of a LUN.
5. Management station sends a request to add space to best utilized thin LUN's. Space will be added to best utilized thin LUN's.
6. The repository also maintains a record of different management stations and the corresponding storage array which can enable or disable for a storage array.

V. Description

In a storage pool find out underutilized LUNs based on the policy parameters provided by the user which can configure the policy parameters either in a configuration file or stored in repository. The policy parameters:

"T" – Time stamp in hours or months or years to determine Utilization of a LUN

"Y" - Space to be freed in percentage of available space.

"X" - %age of LUN's to be picked from the ranking table

"K" – User supplied minimum value to retain free space.

"D" – Dynamic provision flag. If set add the released space to best utilized thin LUNs. If the flag is not set then add the released space to storage pool. "list" – List of LUNs which needs to ranked. Space will be released from underutilized LUNs.

VI. Algorithm

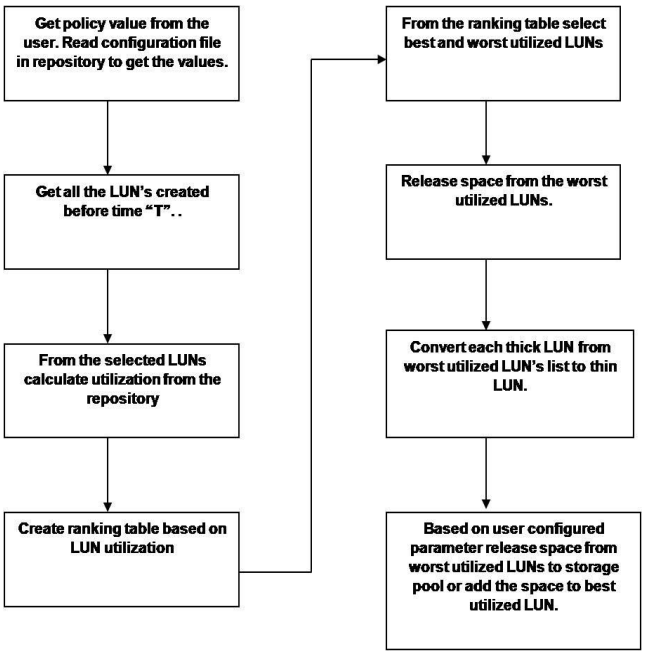

Figure 2. Control flow of proposed algorithm

VII. Release Algorithm

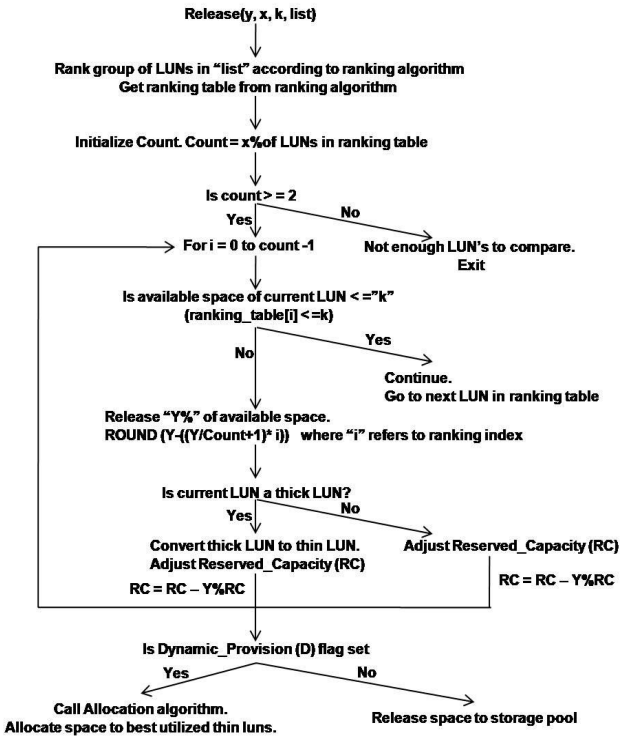

Figure 3. Released algorithm

DOI: 03.LSCS.2013.1.513

VIII. Ranking Algorithm

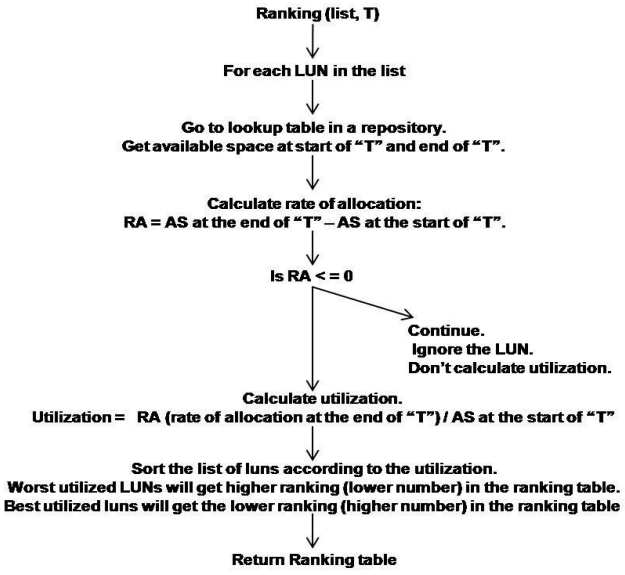

Figure 4. Ranking algorithm

IX. Allocation Algorithm

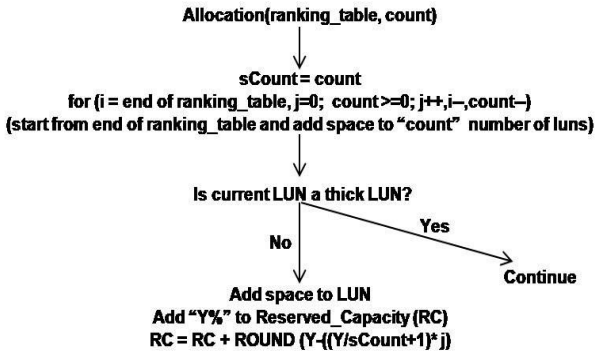

Figure 5. Allocation algorithm

X. Converting From Thick To Thin Volume

In each iteration of the loop, some space will be released from the available (free) space. Once space is released from the available space, thick volume becomes thin volume with new reserved_capacity value.

Reserved_Capacity = available space – y% of available space

XI. Releasing Space From Thin Volume

In each iteration, some space will be released from the available space. Once space is released from the available space, thin volume assumes a new value for reserved_capacity. As for example:

Data in a thick volume need not be contiguous. Free space can be scattered all over the LUN spanning different LBA ranges. In such scenario, free capacity and reserved capacity initially point to available space. Once space is released from the free space reserved capacity will be set to reflect the new value as show in below figure 6.

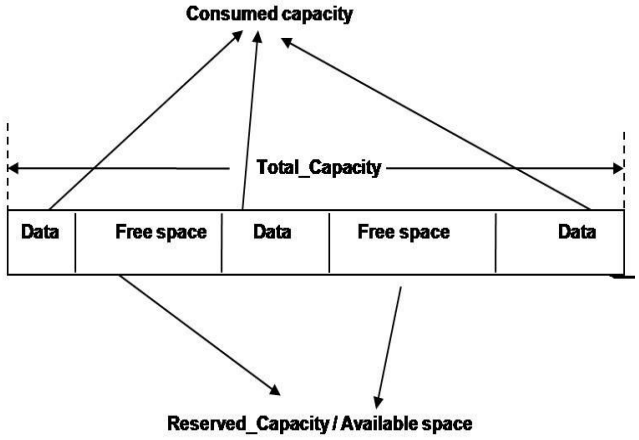

Figure 6. Initial thick volume layout

After releasing some space from free space thick LUN will become thin LUN and it looks as shown in figure 7.

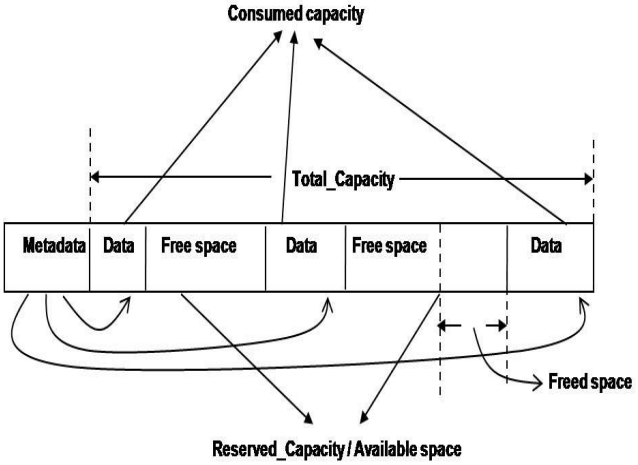

Figure 7. Thick volume layout after releasing space

XII. Releasing Space From Thin Volume

A logical view of initial thin volume layout is shown in figure 8.

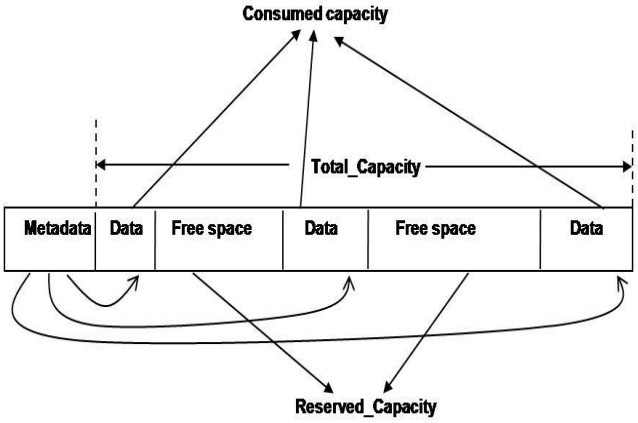

Figure 8. Initial thin volume layout

Logical layout of thin LUN after releasing space from free space will look as shown in figure 9.

It is to be noted that Total_Capacity value doesn't change and hence application like file system won't be having any

DOI: 03.LSCS.2013.1. 513

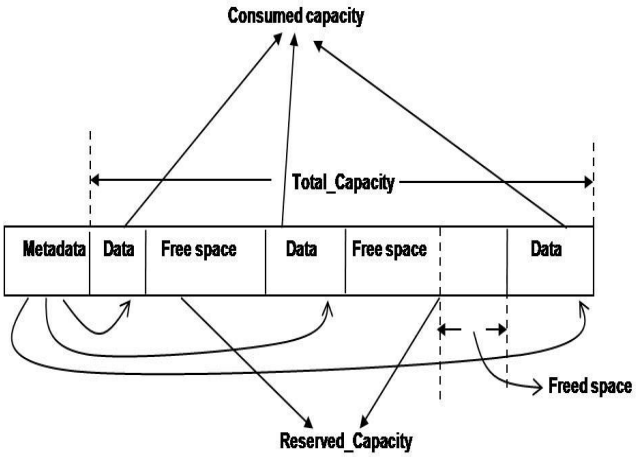

Figure 9. Logical layout of thin LUN after releasing space

impact. All changes will be internal to an intelligent storage system as mentioned Figure 3,4,5.

XIII. Organization Of Look Up Table In A Repository

The Lookup table will be a database and contains information about hourly usage of each LUN in the system as shown in table I.

Table I. Lookup Table

Array Name / ID	LUN ID	Time stamp(hh/dd/mm/yy)	Available space (AS)

The repository maintains a record of different management stations and the corresponding storage array. It's possible to enable or disable storage optimization on a particular storage array managed by management station. The organization of the table is shown in table II.

Table II. Management Station Detail

Management Station	Array Name / ID	Storage optimizer enable/disable

The utilization of a LUN over a period of time "T" can query from the above repository. The difference between available space at the start of time "T" and end of time "T" will give how much space has been used by a LUN.

RA= Rate of allocation, **AS**= Available Space

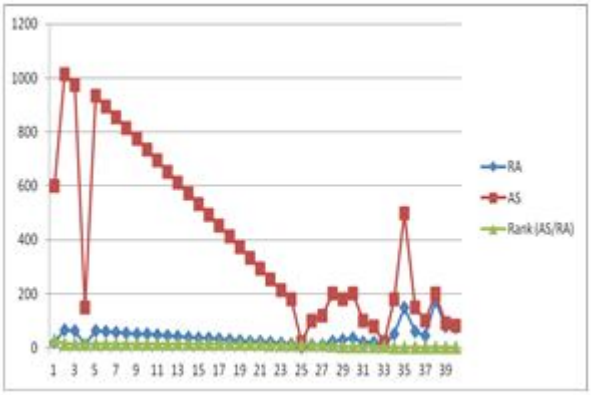

Figure 10. Comparison between lun and the Available space

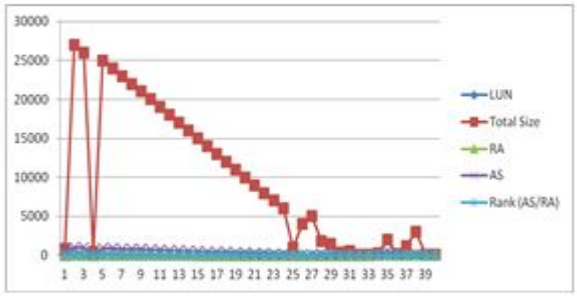

Figure 11. Comparison between LUNs and total space

XIV. Advantages And Disadvantages

The solution has several advantages with significant value add to Business

1. In a cloud environment it is difficult for the storage administrator to do capacity provisioning of each and every storage array. This solution can make use to achieve optimal use of the storage arrays.
2. The proposed solution will be ideal to manage existing storage arrays in a data center optimally.
3. The cost of managing data center will be reduced drastically with the proposed solution. The idea can be extended to manage multiple heterogeneous storage arrays.
4. The cost per TB of the storage can be saved by the solution automatically without compromising the manual operation of the storage array by the administrator in a data center involves cost to the business and is error prone. Also helps in optimal data center growth by optimizing number of storage arrays, space, power and cost per TB.
5. This is the ideal solution for a business relying heavily on many thick LUNs unaware of the underutilized LUNs. It will be cost effective to the business with very minimal or no impact on reliability and performance.

As this solution has several advantages and it has some limitation as well.

1. The solution is best applicable for thick and anchored thin LUNs which are not utilized properly
2. The solution doesn't fit for thin LUNs which are not anchored (or no pre-allocated space) .
1. In the optimal stage of the allocation the solution will have minimal effect.
2. There is an additional cost involved in managing additional resource like repository.
3. Most or all the LUNs will be converted from thick to anchored thin LUN. This might have some performance impact on read or write intensive traffic.

Conclusions

In the storage industry there are many efforts to optimize storage space to save cost which is a value add to the business. The proposed solution can be used to optimize storage space in a complex datacenter configuration or cloud environment. Figure 10 and Figure 11 captures the algorithm result. With the proposed solution it's possible to either dynamically provision LUN's based on utilization or release space from underutilized LUN's to storage pool. Application

DOI: 03.LSCS.2013.1.513

or file system is unaware of underlying storage space optimization in an intelligent storage system and hence there will be minimal or no impact in terms of performance and reliability. The proposed solution can be easily deployed in complex data center or cloud environment to optimally use storage space to save cost.

REFERENCE

[1] Barry L. Wolman and Thomas M. Olson, IOBENCH: A System Independent I0 Benchmark, Computer Architecture News, September, 1989.

[2] David A. Patterson, Garth Gibson, and Randy H. Katz, A Case for Redundant Arrays of Inexpensive Disks (RAID), ACM SIGMOD Conference, Chicago, Illinois,June 1-3, 1988.

[3] E. Anderson, M. Hobbs, K. Keeton, S. Spence, M. Uysal, and A. Veitch, "Hippodrome: Running Circles around Storage Administration," Proc. USENIX Conf. File and Storage Technologies (FAST), pp. 175-188 Jan. 2002.

[4] M. Andrews, M. Bender, and L. Zhang, "New Algorithms for the Disk Scheduling Problem," Proc. IEEE Symp. Foundations of Computer Science, pp. 550-559, Oct. 1996.

[5] O.I. Aven, E.G. Coffman, and Y.A. Kogan, Stochatic Analysis of Computer Storage. D. Reidel, ed., May 1987.

[6] E. Bachmat, "Average Case Analysis for Batched Disk Scheduling and Increasing Subsequences," Proc. 34th Ann. ACM Symp. Theory of Computing, pp. 277-286, May 2002.

DOI: 03.LSCS.2013.1.513